```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>

int counter = 0;  // Shared resource
pthread_mutex_t lock;  // Mutex lock

void* incrementCounter(void* arg) {
    pthread_mutex_lock(&lock);  // Lock the critical section

    for (int i = 0; i < 5; i++) {
        counter++;
        printf("Thread %d: Counter = %d\n", *(int*)arg, counter);
        sleep(1);  // Simulate some work
    }

    pthread_mutex_unlock(&lock);  // Unlock the critical section
    return NULL;
}

int main() {
    pthread_t t1, t2;
    int id1 = 1, id2 = 2;

    pthread_mutex_init(&lock, NULL);  // Initialize mutex

    // Create threads
    pthread_create(&t1, NULL, incrementCounter, &id1);
```

```
    pthread_create(&t2, NULL, incrementCounter, &id2);

    // Wait for threads to finish
    pthread_join(t1, NULL);
    pthread_join(t2, NULL);

    pthread_mutex_destroy(&lock);  // Destroy mutex

    printf("Final Counter Value: %d\n", counter);

    return 0;
}
```